

[illegible]

```
IIIIII  NN  NN  IIIII  FFFFFFFF  CCCCCC  222222
IIIIII  NN  NN  IIIII  FFFFFFFF  CCCCCC  222222
  II    NN  NN  II     FF          CC          22  22
  II    NN  NN  II     FF          CC          22  22
  II    NN  NN  II     FF          CC          22  22
  II    NNNN NN  II     FF          CC          22  22
  II    NNNN NN  II     FF          CC          22  22
  II    NN  NN  II     FFFFFFFF  CC          22
  II    NN  NN  II     FFFFFFFF  CC          22
  II    NN  NN  II     FF          CC          22
  II    NN  NN  II     FF          CC          22
  II    NN  NN  II     FF          CC          22
  II    NN  NN  II     FF          CC          22
IIIIII  NN  NN  IIIII  FF          CCCCCC  2222222222
IIIIII  NN  NN  IIIII  FF          CCCCCC  2222222222
```

```
LL      IIIII  SSSSSSSS
LL      IIIII  SSSSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SSSSSS
LL      II     SSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LLLLLLLL  IIIII  SSSSSSSS
LLLLLLLL  IIIII  SSSSSSSS
```



```
1 0001 0 MODULE INIFC2 (
2 0002 0 LANGUAGE (BLISS32),
3 0003 0 IDENT = 'V04-000',
4 0004 0 ) =
5 0005 1 BEGIN
6 0006 1
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: F11ACP Structure Level 2
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 These routines create and initialize a file control block
38 0038 1 from the given file header.
39 0039 1
40 0040 1 ENVIRONMENT:
41 0041 1
42 0042 1 STARLET operating system, including privileged system services
43 0043 1 and internal exec routines. These routines must be called in
44 0044 1 kernel mode.
45 0045 1
46 0046 1
47 0047 1 --
48 0048 1
49 0049 1
50 0050 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 14-Dec-1976 16:48
51 0051 1
52 0052 1 MODIFIED BY:
53 0053 1
54 0054 1 V03-015 CDS0006 Christian D. Saether 30-Aug-1984
55 0055 1 Add optional PRIMFCB argument to INIT_FCB2 and
56 0056 1 FILL_FCB routines.
57 0057 1
```

58	0058	1	V03-014	CDS0005	Christian D. Saether	15-Aug-1984	
59	0059	1			Set MARKDEL and BADBLOCK flags from header.		
60	0060	1					
61	0061	1	V03-013	CDS0004	Christian D. Saether	31-July-1984	
62	0062	1			Remove local declaration of get_map_pointer linkage.		
63	0063	1			Use locals to reference fcb and header arguments.		
64	0064	1			Remove test for directory fcb.		
65	0065	1			Change acl insque to add at end of list.		
66	0066	1			Use common acl init routine.		
67	0067	1					
68	0068	1	V03-012	ACG0440	Andrew C. Goldstein,	23-Jul-1984 11:56	
69	0069	1			Add ref count and classification valid flag to ORB in the FCB		
70	0070	1					
71	0071	1	V03-011	LMP0275	L. Mark Pilant,	12-Jul-1984 15:08	
72	0072	1			Initialize the ACL info in the ORB to be a null descriptor		
73	0073	1			list rather than an empty queue. This avoids the overhead		
74	0074	1			of locking and unlocking the ACL mutex, only to find out		
75	0075	1			that the ACL was empty.		
76	0076	1					
77	0077	1	V03-010	CDS0003	Christian D. Saether	26-Apr-1984	
78	0078	1			Set up FCB\$_LOCKBASIS.		
79	0079	1					
80	0080	1	V03-009	LMP0221	L. Mark Pilant,	7-Apr-1984 11:58	
81	0081	1			Add support for the ORB within the FCB.		
82	0082	1					
83	0083	1	V03-008	ACG0408	Andrew C. Goldstein,	20-Mar-1984 17:53	
84	0084	1			Make APPLY_RVN and DEFAULT_RVN macros		
85	0085	1					
86	0086	1	V03-007	LMP0208	L. Mark Pilant,	9-Mar-1984 9:21	
87	0087	1			Copy the BADACL bit from the header to the FCB.		
88	0088	1					
89	0089	1	V03-006	LMP0188	L. Mark Pilant,	3-Feb-1984 15:45	
90	0090	1			Fill in the classification block in the FCB if it exists		
91	0091	1			in the file header.		
92	0092	1					
93	0093	1	V03-005	CDS0002	Christian D. Saether	18-Jan-1984	
94	0094	1			Modify interface to APPLY_RVN.		
95	0095	1					
96	0096	1	V03-004	CDS0001	Christian D. Saether	30-Dec-1983	
97	0097	1			Use L_NORM linkage and BIND_COMMON macro.		
98	0098	1					
99	0099	1	V03-003	LMP0073	L. Mark Pilant,	20-Jan-1983 9:31	
100	0100	1			Use the reserved area offset as the upper limit to the		
101	0101	1			ACL segment size in a file header.		
102	0102	1					
103	0103	1	V03-002	ACG0308	Andrew C. Goldstein,	14-Jan-1983 15:21	
104	0104	1			Fix FCB linkage consistency problems		
105	0105	1					
106	0106	1	V03-001	LMP0068	L. Mark Pilant,	21-Dec-1982 12:11	
107	0107	1			If the header contains ACL information, create a new ACL		
108	0108	1			segment and link it into the chain.		
109	0109	1					
110	0110	1	V02-003	ACG0241	Andrew C. Goldstein,	11-Dec-1981 22:53	
111	0111	1			Make updating of FCB common code, add handling		
112	0112	1			of directory bit		
113	0113	1					
114	0114	1	V02-002	ACG0167	Andrew C. Goldstein,	16-Apr-1980 19:26	

INIFC2
V04-000

N 15
16-Sep-1984 00:36:47
14-Sep-1984 12:30:31

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11X.SRC]INIFC2.B32;1 Page 3 (1)

Previous revision history moved to F11B.REV

:	115	0115	1	!	
:	116	0116	1	!*	
:	117	0117	1		
:	118	0118	1		
:	119	0119	1	LIBRARY	'SYSS\$LIBRARY:LIB,L32';
:	120	0120	1	REQUIRE	'SRC\$:FCPDEF.B32';
:	121	1111	1		
:	122	1112	1		
:	123	1113	1	FORWARD ROUTINE	
:	124	1114	1	INIT_FCB2	: L_NORM NOVALUE, ! initialize FCB
:	125	1115	1	FILL_FCB	: L_NORM NOVALUE; ! fill in basic FCB fields

BCDEFGHIJKLMNOPQRSTUVWXYZBCDEFGHIJKLMNOPQRSTUVWXYZBCDEFGHIJKLMNOPQRSTUVWXYZ


```
1116 1 GLOBAL ROUTINE INIT_FCB2 (FCBARG, HEADER, PRIMFCB) : L_NORM NOVALUE =
1117 1
1118 1 ++
1119 1
1120 1 FUNCTIONAL DESCRIPTION:
1121 1
1122 1     This routine initializes the given FCB according to the given
1123 1     file header.
1124 1
1125 1 CALLING SEQUENCE:
1126 1     INIT_FCB (ARG1, ARG2)
1127 1
1128 1 INPUT PARAMETERS:
1129 1     ARG1: FCB address
1130 1     ARG2: header address
1131 1     ARG3: optional arg to specify primary fcb
1132 1
1133 1 IMPLICIT INPUTS:
1134 1     HEADER_LBN contains LBN of header block
1135 1
1136 1 OUTPUT PARAMETERS:
1137 1     NONE
1138 1
1139 1 IMPLICIT OUTPUTS:
1140 1     NONE
1141 1
1142 1 ROUTINE VALUE:
1143 1     NONE
1144 1
1145 1 SIDE EFFECTS:
1146 1     FCB initialized
1147 1
1148 1 --
1149 1
1150 2 BEGIN
1151 2
1152 2 MAP
1153 2     HEADER          : REF BBLOCK;    ! file header arg
1154 2
1155 2 GLOBAL REGISTER
1156 2     COUNT           = 6,             ! retrieval pointer count
1157 2     LBN             = 7,             ! retrieval pointer LBN
1158 2     MAP_POINTER     = 8;             ! pointer to scan map area
1159 2
1160 2 BASE_REGISTER;
1161 2
1162 2 EXTERNAL ROUTINE
1163 2     GET_MAP_POINTER : L_MAP_POINTER; ! get value of file map pointer
1164 2
1165 2 LOCAL
1166 2     FCB              : REF BBLOCK;
1167 2
1168 2 FCB = .FCBARG;
1169 2
1170 2 ! Scan the map area. Count up the file size from the retrieval pointers.
1171 2 !
1172 2
```



```
184 1173 2 MAP_POINTER = .HEADER + .HEADER[FH2$B_MPOFFSET]*2;
185 1174 2 FCB[FCB$L_FILESIZE] = 0;
186 1175 2
187 1176 2 UNTIL .MAP_POINTER GEQA .HEADER + (.HEADER[FH2$B_MPOFFSET] + .HEADER[FH2$B_MAP_INUSE]) * 2
188 1177 2 DO
189 1178 2 BEGIN
190 1179 2 GET_MAP_POINTER ();
191 1180 2 FCB[FCB$L_FILESIZE] = .FCB[FCB$L_FILESIZE] + .COUNT;
192 1181 2 END;
193 1182 2
194 1183 2 ! Now set up the rest of the fields.
195 1184 2 !
196 1185 2
197 1186 2 IF ACTUALCOUNT EQL 3
198 1187 2 THEN
199 1188 2 FILL_FCB (.FCB, .HEADER, .PRIMFCB)
200 1189 2 ELSE
201 1190 2 FILL_FCB (.FCB, .HEADER);
202 1191 2
203 1192 2
204 1193 1 END;

! end of routine INIT_FCB
```

				.TITLE	INIFC2	
				.IDENT	\V04-000\	
				.EXTRN	GET_MAP_POINTER	
				.PSECT	\$CODE\$,NOWRT,2	
				.ENTRY	INIT_FCB2, Save R2,R3,R6,R7,R8	1116
52	04	AC	D0 00002	MOVL	FCBARG, FCB	1168
51	08	AC	D0 00006	MOVL	HEADER, R1	1173
50	01	A1	9A 0000A	MOVZBL	1(R1), R0	
58		6140	3E 0000E	MOVAW	(R1)[R0], MAP_POINTER	
	38	A2	D4 00012	CLRL	56(FCB)	1174
51	08	AC	D0 00015	MOVL	HEADER, R1	1176
50	01	A1	9A 00019	MOVZBL	1(R1), R0	
53	3A	A1	9A 0001D	MOVZBL	58(R1), R3	
50		53	C0 00021	ADDL2	R3, R0	
50		6140	3E 00024	MOVAW	(R1)[R0], R0	
50		58	D1 00028	CMPL	MAP_POINTER, R0	
		09	1E 0002B	BGEQU	2\$	
		0000G	30 0002D	BSBW	GET_MAP_POINTER	1179
38	A2	56	C0 00030	ADDL2	COUNT, 56(FCB)	1180
		DF	11 00034	BRB	1\$	1176
03		6C	91 00036	CMPB	(AP), #3	1186
		0C	12 00039	BNEQ	3\$	
7E	08	AC	7D 0003B	MOVQ	HEADER, -(SP)	1188
		52	DD 0003F	PUSHL	FCB	
0000V	CF	03	FB 00041	CALLS	#3, FILL_FCB	
			04 00046	RET		
	08	AC	DD 00047	PUSHL	HEADER	1190
		52	DD 0004A	PUSHL	FCB	
0000V	CF	02	FB 0004C	CALLS	#2, FILL_FCB	
			04 00051	RET		1193

INIFC2
V04-000

D 16
16-Sep-1984 00:36:47
14-Sep-1984 12:30:31

VAX-11 Bliss-32 V4.0-742
DISK\$VMMASTER:[F11X.SRC]INIFC2.B32;1 Page 6 (2)

; Routine Size: 82 bytes, Routine Base: \$CODE\$ + 0000


```
206 1194 1 GLOBAL ROUTINE FILL_FCB (FCBARG, HDRARG, PRIMFCB) : L_NORM NOVALUE =
207 1195 1
208 1196 1 ++
209 1197 1
210 1198 1 FUNCTIONAL DESCRIPTION:
211 1199 1
212 1200 1 This routine updates the file attributes of the specified FCB,
213 1201 1 if any, with the file attributes of the given header. The file size
214 1202 1 is preserved.
215 1203 1
216 1204 1
217 1205 1 CALLING SEQUENCE:
218 1206 1 FILL_FCB (ARG1, ARG2)
219 1207 1
220 1208 1 INPUT PARAMETERS:
221 1209 1 ARG1: address of FCB
222 1210 1 ARG2: address of file header
223 1211 1 ARG3: optional arg to specify primary fcb
224 1212 1
225 1213 1 IMPLICIT INPUTS:
226 1214 1 NONE
227 1215 1
228 1216 1 OUTPUT PARAMETERS:
229 1217 1 NONE
230 1218 1
231 1219 1 IMPLICIT OUTPUTS:
232 1220 1 NONE
233 1221 1
234 1222 1 ROUTINE VALUE:
235 1223 1 NONE
236 1224 1
237 1225 1 SIDE EFFECTS:
238 1226 1 FCB is updated if it exists
239 1227 1
240 1228 1 --
241 1229 1
242 1230 2 BEGIN
243 1231 2
244 1232 2 BUILTIN
245 1233 2 REMQUE;
246 1234 2
247 1235 2 LOCAL
248 1236 2 FCB : REF BBLOCK, ! local copy of fcb address
249 1237 2 FCB_ORB : REF BBLOCK, ! Address of the ORB within the FCB
250 1238 2 HEADER : REF BBLOCK, ! local copy of header arg
251 1239 2 ACE_ADDRESS : REF BBLOCK, ! Address of current ACE in header
252 1240 2 ACL_LENGTH, ! Size of ACL segment to create
253 1241 2 ACL_ADDRESS : REF BBLOCK, ! Address of created ACL segment
254 1242 2 ACL_FCB : REF BBLOCK, ! FCB for ACL segment
255 1243 2 DUMMY; ! dummy target for REMQUE
256 1244 2
257 1245 2 GLOBAL REGISTER
258 1246 2 COUNT = 6, ! retrieval pointer count
259 1247 2 LBN = 7, ! retrieval pointer LBN
260 1248 2 MAP_POINTER = 8; ! pointer to scan map area
261 1249 2
262 1250 2 BIND_COMMON;
```



```
263 1251 2
264 1252 2
265 1253 2 EXTERNAL
266 1254 2     EXESGL_DYNAMIC_FLAGS      : BITVECTOR ADDRESSING MODE (ABSOLUTE),
267 1255 2     CTL$GL_PCB              : REF BBLOCK ADDRESSING MODE (GENERAL);
268 1256 2     ! Dynamic SYSGEN flags
269 1257 2     ! PCB address
270 1258 2 EXTERNAL LITERAL
271 1259 2     EXESV_CLASS_PROT;          ! Set if doing non-discretionary checks
272 1260 2
273 1261 2 EXTERNAL ROUTINE
274 1262 2     ACL_INIT_QUEUE,            ! ACL initialization
275 1263 2     GET_MAP_POINTER : L_MAP_POINTER, ! get value of file map pointer
276 1264 2     ALLOC_PAGED      : L_NORM;      ! Allocate system dynamic memory
277 1265 2
278 1266 2 ! Note the address of the ORB within the FCB.
279 1267 2
280 1268 2 FCB = .FCBARG;
281 1269 2 HEADER = .HDRARG;
282 1270 2 FCB_ORB = FCB[FCB$R_ORB];
283 1271 2
284 1272 2 ! Get the known constants and the simple stuff from the file header
285 1273 2 ! (i.e., header LBN, file ID, starting VBN, file owner and file protection).
286 1274 2 !
287 1275 2
288 1276 2 FCB[FCB$L_HDLBN] = .HEADER_LBN;
289 1277 2 FCB[FCB$W_FID_NUM] = .HEADER[FH2$W_FID_NUM];
290 1278 2 FCB[FCB$W_FID_SEQ] = .HEADER[FH2$W_FID_SEQ];
291 1279 2 FCB[FCB$W_FID_RVN] = .HEADER[FH2$W_FID_RVN];
292 1280 2 FCB[FCB$W_SEG_N] = .HEADER[FH2$W_SEG_NUM];
293 1281 2 FCB_ORB[ORB$L_OWNER] = .HEADER[FH2$L_FILEOWNER];
294 1282 2 FCB_ORB[ORB$V_PROT_16] = 1;
295 1283 2 FCB_ORB[ORB$W_PROT] = .HEADER[FH2$W_FILEPROT];
296 1284 2 FCB_ORB[ORB$V_MODE_VECTOR] = 1;
297 1285 2 FCB_ORB[ORB$L_MODE_PROTL] = NOT .HEADER[FH2$B_ACC_MODE];
298 1286 2 FCB_ORB[ORB$L_MODE_PROTH] = -1;
299 1287 2
300 1288 2 $ASSUME (FH2$S_CLASS_PROT EQL ORB$S_MIN_CLASS);
301 1289 2 $ASSUME (FH2$S_CLASS_PROT EQL ORB$S_MAX_CLASS);
302 1290 2
303 1291 2 IF .EXESGL_DYNAMIC_FLAGS[EXESV_CLASS_PROT]
304 1292 2 AND .HEADER[FH2$B_IDOFFSET]*2 GTRU FH2$C_LENGTH
305 1293 2 THEN
306 1294 2     BEGIN
307 1295 2     CH$MOVE (FH2$S_CLASS_PROT, HEADER[FH2$R_CLASS_PROT], FCB_ORB[ORB$R_MIN_CLASS]);
308 1296 2     CH$MOVE (FH2$S_CLASS_PROT, HEADER[FH2$R_CLASS_PROT], FCB_ORB[ORB$R_MAX_CLASS]);
309 1297 2     FCB_ORB[ORB$V_CLASS_PROT] = 1;
310 1298 2     END;
311 1299 2
312 1300 2 FCB[FCB$W_VERSIONS] = .BBLOCK [HEADER[FH2$W_RECATTR], FAT$W_VERSIONS];
313 1301 2 APPLY RVN (FCB[FCB$W_FID_RVN], .CURRENT_RVN);
314 1302 2 (FCB[FCB$L_LOCKBASIS]<0,16> = .FCB[FCB$W_FID_NUM];
315 1303 2 (FCB[FCB$L_LOCKBASIS]<16,8> = .FCB[FCB$B_FID_NMX];
316 1304 2 (FCB[FCB$L_LOCKBASIS]<24,8> = .FCB[FCB$B_FID_RVN];
317 1305 2
318 1306 2 IF .HEADER[FH2$V_SPOOL] THEN FCB[FCB$V_SPOOL] = 1;
319 1307 2 IF .HEADER [FH2$V_BADBLOCK] THEN FCB [FCB$V_BADBLK] = 1;
```



```

320 1308 2 IF .HEADER [FH2$V_MARKDEL] THEN FCB [FCB$V_MARKDEL] = 1;
321 1309 2
322 1310 2 FCB[FCB$L_EFBLK] = ROT (.BBLOCK[HEADER[FH2$W_RECATTR], FAT$L_EFBLK], 16);
323 1311 2 IF .FCB[FCB$L_EFBLK] NEQ 0
324 1312 2 AND .BBLOCK[HEADER[FH2$W_RECATTR], FAT$W_FFBYTE] EQL 0
325 1313 2 THEN FCB[FCB$L_EFBLK] = .FCB[FCB$L_EFBLK] - 1;
326 1314 2
327 1315 2 IF .FCB[FCB$L_EFBLK] GTR .FCB[FCB$L_FILESIZE]
328 1316 2 THEN FCB[FCB$L_EFBLK] = .FCB[FCB$L_FILESIZE];
329 1317 2
330 1318 2 ! Now scan the map area. Get the starting LBN if the file is contiguous.
331 1319 2 !
332 1320 2
333 1321 2 MAP_POINTER = .HEADER + .HEADER[FH2$B_MPOFFSET]*2;
334 1322 2 GET_MAP_POINTER ();
335 1323 2 FCB[FCB$L_STLBN] = 0;
336 1324 2 IF .HEADER[FH2$V_CONTIG]
337 1325 2 THEN FCB[FCB$L_STLBN] = .LBN;
338 1326 2
339 1327 2 ! Build an ACL segment if ACL information exists in the file header,
340 1328 2 ! and this is not an FCB update.
341 1329 2
342 1330 2 IF .HEADER[FH2$V_BADACL] THEN FCB[FCB$V_BADACL] = 1;
343 1331 2
344 1332 2 IF .HEADER[FH2$B_ACOFFSET]*2 LSS .HEADER[FH2$B_RSOFFSET]*2
345 1333 2 AND NOT .CLEANUP_FLAGS[CLF_NOBUILD]
346 1334 2 THEN
347 1335 3 BEGIN
348 1336 3 ACE_ADDRESS = .HEADER + .HEADER[FH2$B_ACOFFSET]*2;
349 1337 3 ACL_LENGTH = 0;
350 1338 3 DO
351 1339 4 BEGIN
352 1340 4 ACL_LENGTH = .ACL_LENGTH + .ACE_ADDRESS[ACE$B_SIZE];
353 1341 4 ACE_ADDRESS = .ACE_ADDRESS + .ACE_ADDRESS[ACE$B_SIZE];
354 1342 4 END
355 1343 3 UNTIL .ACE_ADDRESS[ACE$B_SIZE] EQL 0
356 1344 3 OR .ACE_ADDRESS GEQA .HEADER + .HEADER[FH2$B_RSOFFSET]*2;
357 1345 3 IF .ACL_LENGTH GTR 0
358 1346 3 THEN
359 1347 4 BEGIN
360 1348 4 ACL_ADDRESS = ALLOC_PAGED (ACL$C_LENGTH + .ACL_LENGTH, ACL_TYPE);
361 1349 4 CH$MOVE (.ACL_LENGTH, .HEADER + .HEADER[FH2$B_ACOFFSET]*2,
362 1350 4 ACL_ADDRESS[ACL$L_LIST]);
363 1351 4 ACL_ADDRESS[ACL$W_SIZE] = ACL$C_LENGTH + .ACL_LENGTH;
364 1352 4
365 1353 4 IF .FCB [FCB$W_SEGN] EQL 0
366 1354 4 THEN
367 1355 4 ACL_FCB = .FCB
368 1356 4 ELSE
369 1357 4 IF ACTUALCOUNT EQL 3
370 1358 4 THEN
371 1359 4 ACL_FCB = .PRIMFCB
372 1360 4 ELSE
373 1361 4 ACL_FCB = .PRIMARY_FCB;
374 1362 4
375 1363 4 FCB_ORB = ACL_FCB[FCB$R_ORB];
376 1364 4 IF NOT .FCB_ORB[ORB$V_ACL_QUEUE]
```



```

: 377      1365 4      THEN
: 378      1366 4      ACL_INIT_QUEUE (.FCB_ORB);
: 379      1367 4
: 380      1368 4      INSQUE (.ACL_ADDRESS, .FCB_ORB[ORB$L_ACLBL]);
: 381      1369 4
: 382      1370 3      END;
: 383      1371 2      END;
: 384      1372 2      CLEANUP_FLAGS[CLF_NOBUILD] = 0;
: 385      1373 2
: 386      1374 1      END;

! end of routine FILL_FCB
```

```

                                .EXTRN  EXESGL_DYNAMIC_FLAGS
                                .EXTRN  CTL$GL_PCB, EXESV_CLASS_PROT
                                .EXTRN  ACL_INIT_QUEUE, ACLOC_PAGED

                                .ENTRY  FILL_FCB, Save R2,R3,R4,R5,R6,R7,R8,R9,R11
                                : 1194
                                MOVL    FCBARG, FCB
                                : 1268
                                MOVL    HDRARG, HEADER
                                : 1269
                                PUSHAB  88(R9)
                                : 1270
                                MOVL    -80(BASE), 52(FCB)
                                : 1276
                                MOVL    8(HEADER), 36(FCB)
                                : 1277
                                MOVW    12(HEADER), 40(FCB)
                                : 1279
                                MOVW    4(HEADER), 42(FCB)
                                : 1280
                                MOVL    60(HEADER), @FCB_ORB
                                : 1281
                                ADDL3   #11, FCB_ORB, R0
                                : 1282
                                BISB2   #1, (R0)
                                :
                                ADDL3   #24, FCB_ORB, R0
                                : 1283
                                MOVW    64(HEADER), (R0)
                                :
                                ADDL3   #11, FCB_ORB, R0
                                : 1284
                                BISB2   #4, (R0)
                                :
                                ADDL3   #16, FCB_ORB, R0
                                : 1285
                                MOVZBL  59(HEADER), (R0)
                                :
                                ADDL3   #16, FCB_ORB, R1
                                :
                                ADDL3   #16, FCB_ORB, R2
                                :
                                MCOML   (R2), (RT)
                                :
                                ADDL3   #20, FCB_ORB, R0
                                : 1286
                                MNEGL   #1, (R0)
                                :
                                BBC     #EXESV_CLASS_PROT, @#EXESGL_DYNAMIC_FLAGS, -
                                : 1291
                                :
                                MOVZBL  (HEADER), R1
                                : 1292
                                MULL2   #2, R1
                                :
                                CMPL   R1, #80
                                :
                                BLEQU   1$,
                                :
                                ADDL3   #48, FCB_ORB, R6
                                : 1295
                                MOVW    #20, 88(HEADER), (R6)
                                :
                                ADDL3   #68, FCB_ORB, R6
                                : 1296
                                MOVW    #20, 88(HEADER), (R6)
                                :
                                ADDL3   #11, FCB_ORB, R0
                                : 1297
                                BISB2   #16, (R0)
                                :
                                MOVW    50(HEADER), 64(FCB)
                                : 1300
                                TSTB    40(FCB)
                                : 1301
                                BNEQ    2$,
                                :
                                MOVW    -96(BASE), 40(FCB)
                                :
                                CMPB   40(FCB), #1
                                :
                                BNEQ    3$,
                                :
```


			A0	AA	D5	000A3	TSTL	-96(BASE)	:	
				03	12	000A6	BNEQ	3\$	:	
			28	A9	94	000A8	CLRB	40(FCB)	:	
		4C	24	A9	B0	000AB	MOVW	36(FCB), 76(FCB)	:	1302
		4E	29	A9	90	000B0	MOVB	41(FCB), 78(FCB)	:	1303
		4F	28	A9	90	000B5	MOVB	40(FCB), 79(FCB)	:	1304
04		35		04	E1	000BA	BBC	#4, 53(HEADER), 4\$	:	1306
		22		10	88	000BF	BISB2	#16, 34(FCB)	:	
04		35		06	E1	000C3	BBC	#6, 53(HEADER), 5\$	:	1307
		22		04	88	000C8	BISB2	#4, 34(FCB)	:	
			35	AB	95	000CC	TSTB	53(HEADER)	:	1308
				04	18	000CF	BGEQ	6\$	:	
		22		02	88	000D1	BISB2	#2, 34(FCB)	:	
3C	A9	1C		10	9C	000D5	ROTL	#16, 28(HEADER), 60(FCB)	:	1310
				08	13	000DB	BEQL	7\$	:	1311
			20	AB	B5	000DD	TSTW	32(HEADER)	:	1312
				03	12	000E0	BNEQ	7\$	:	
		38		A9	D7	000E2	DECL	60(FCB)	:	1313
			3C	A9	D1	000E5	CMPL	60(FCB), 56(FCB)	:	1315
				05	15	000EA	BLEQ	8\$	:	
		3C		A9	D0	000EC	MOVL	56(FCB), 60(FCB)	:	1316
			01	AB	9A	000F1	MOVZBL	1(HEADER), R0	:	1321
				6B40	3E	000F5	MOVAW	(HEADER)[R0], MAP_POINTER	:	
				0000G	30	000F9	BSBW	GET MAP_POINTER	:	1322
			30	A9	D4	000FC	CLRL	48(FCB)	:	1323
			34	AB	95	000FF	TSTB	52(HEADER)	:	1324
				04	18	00102	BGEQ	9\$	:	
		30		57	D0	00104	MOVL	LBN, 48(FCB)	:	1325
05		35		03	E1	00108	BBC	#3, 53(HEADER), 10\$	:	1330
		22		80	8F	0010D	BISB2	#128, 34(FCB)	:	
			02	AB	9A	00112	MOVZBL	2(HEADER), R1	:	1332
				02	C4	00116	MULL2	#2, R1	:	
			03	AB	9A	00119	MOVZBL	3(HEADER), R0	:	
				02	C4	0011D	MULL2	#2, R0	:	
				51	D1	00120	CMPL	R1, R0	:	
				76	18	00123	BGEQ	16\$	:	
72				0B	E0	00125	BBS	#11, (BASE), 16\$	:	1333
		6A		5B	C0	00129	ADDL2	HEADER, ACE_ADDRESS	:	1336
		51		57	D4	0012C	CLRL	ACL_LENGTH	:	1337
				61	9A	0012E	MOVZBL	(ACE_ADDRESS), R0	:	1340
		50		50	C0	00131	ADDL2	R0, ACL_LENGTH	:	
		57		61	9A	00134	MOVZBL	(ACE_ADDRESS), R0	:	1341
		50		50	C0	00137	ADDL2	R0, ACE_ADDRESS	:	
		51		61	95	0013A	TSTB	(ACE_ADDRESS)	:	1343
				0D	13	0013C	BEQL	12\$	:	
			03	AB	9A	0013E	MOVZBL	3(HEADER), R0	:	1344
		50		6B40	3E	00142	MOVAW	(HEADER)[R0], R0	:	
		50		51	D1	00146	CMPL	ACE_ADDRESS, R0	:	
				E3	1F	00149	BLSSU	11\$	:	
				57	D5	0014B	TSTL	ACL_LENGTH	:	1345
				4C	15	0014D	BLEQ	16\$	:	
				07	DD	0014F	PUSHL	#7	:	1348
			0C	A7	9F	00151	PUSHAB	12(ACL_LENGTH)	:	
				02	FB	00154	CALLS	#2, ALLOC PAGED	:	
				50	D0	00159	MOVL	R0, ACL_ADDRESS	:	
			02	AB	9A	0015C	MOVZBL	2(HEADER), R0	:	1349
				6B40	3F	00160	PUSHAW	(HEADER)[R0]	:	1350

INIFC2
V04-000

J 16
16-Sep-1984 00:36:47
14-Sep-1984 12:30:31

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11X.SRC]INIFC2.B32;1

Page 12
(3)

0C	A6	9E	57	28	00163	MOV C3	ACL_LENGTH, @ (SP) + 12 (ACL_ADDRESS)	:	1351	
08	A6	57	0C	A1	00168	ADD W3	#12, ACL_LENGTH, 8 (ACL_ADDRESS)	:	1351	
			2A	A9	B5	0016D	TST W	42 (FCB)	:	1353
				0F	13	00170	BEQ L	14\$	:	
		03	6C	91	00172	CMP B	(AP), #3	:	1357	
			06	12	00175	BNEQ	13\$	:		
		59	0C	AC	D0	00177	MOVL	PRIMFCB, ACL_FCB	:	1359
			04	11	0017B	BRB	14\$	:		
		59	08	AA	D0	0017D	13\$: MOVL	8 (BASE), ACL_FCB	:	1361
		6E	58	A9	9E	00181	14\$: MOVAB	88 (R9), FCB_ORB	:	1363
	50	6E		0B	C1	00185	ADD L3	#11, FCB_ORB, R0	:	1364
	07	60		01	E0	00189	BBS	#1, (R0), 15\$	:	
				6E	DD	0018D	PUSHL	FCB_ORB	:	1366
		0000G	CF	01	FB	0018F	CALLS	#1, ACL_INIT_QUEUE	:	
	50	6E		2C	C1	00194	15\$: ADD L3	#4, FCB_ORB, R0	:	1368
		90		66	0E	00198	INSQUE	(ACL_ADDRESS), @ (R0) +	:	
		01	AA	08	8A	0019B	16\$: BIC B2	#8, T (BASE)	:	1372
				04	0019F	RET		:	1374	

; Routine Size: 416 bytes, Routine Base: \$CODE\$ + 0052

: 387 1375 1
: 388 1376 1 END
: 389 1377 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	498	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	76	0	1000	00:02.0

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD, INITIAL, OPTIMIZE)/LIS=LIS\$:INIFC2/OBJ=OBJ\$:INIFC2 MSRC\$:INIFC2/UPDATE=(ENH\$:INIFC2)

INIFC2
V04-000

K 16
16-Sep-1984 00:36:47

VAX-11 Bliss-32 V4.0-742

Page 13

: Size: 498 code + 0 data bytes
: Run Time: 00:24.9
: Elapsed Time: 00:44.8
: Lines/CPU Min: 3318
: Lexemes/CPU-Min: 41508
: Memory Used: 311 pages
: Compilation Complete

0170

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY